An Architecture Proposal for the ILC Test Beam Silicon Telescope at Fermilab

M.Turqueti, R.Rivera, L.Uplegger, A.Prosser, *Fermi National Accelerator Laboratory, Batavia,IL, USA*

Abstract- The requirements for an ILC Test Beam silicon telescope system are foreseen to be very stringent. Resolution, noise, and throughput must be carefully managed in order to provide a useful instrument for the high energy physics community to develop detector technologies for the ILC. Since the ILC Test Beam is meant to test a wide variety of different detectors, it must employ universally accepted software techniques, hardware standards and protocols as well as easy integration of hardware and software with the various clients using the system.

In this paper, we describe an open modular architecture to achieve these goals, including an analysis of the entire chain of software and hardware needed to meet the requirements.

I. Introduction

The design of the silicon telescope architecture is crucial to the construction of a reliable, user-friendly and effective test facility to develop the ILC tracker detector. Choosing the right strategy for the development of the silicon telescope is essential to avoid serious problems with the future ILC test beam program.

The design of such a telescope system offers several challenges. First, it must support the predicted data rate to which the system will be exposed. Second, it must be flexible to deal with changes in the requirements such as the number of sensors being read out. Third, the DAQ must be able to deal with a wide variety of detectors that will be tested; therefore the integration of these devices with the readout system must be straightforward.

This work provides an initial overview of the chosen architecture, analyzing its main components and protocols.

The system implementation is then illustrated as well as the capabilities of the system and the possible upgrades.

Finally, a summary of the system capabilities is presented and explained.

Manuscript received April 27, 2007. This work was supported by the U.S. Department of Energy Operated by Fermi Research Alliance, LLC under Contract No.DE-AC02-07CH11359 with the United States Department of Energy.

Marcos Turqueti is with the Fermi National Accelerator Laboratory, PO Box 500 Batavia, IL 50510 (e-mail: turqueti@fnal.gov).

Ryan Rivera is with the Fermi National Accelerator Laboratory, PO Box 500 Batavia, IL 50510 (e-mail: rrivera@fnal.gov).

Lorenzo Uplegger is with the Fermi National Accelerator Laboratory, PO Box 500 Batavia, IL 50510 (e-mail: turqueti@fnal.gov).

Alan Prosser is with the Fermi National Accelerator Laboratory, PO Box 500 Batavia, IL 50510 (e-mail: aprosser@fnal.gov).

This system is a new telescope DAQ with the goal of replacing the previous DAQ built for BTeV/CMS [1] [2].

II. The Network Based Solution

The basic idea of the proposed architecture is to provide a LAN like network Ethernet/IP based solution [3]. In this system, the telescope and the detector under test (DUT) are lodged on stations that have built-in networking capabilities. There are three types of stations, the X and Y telescope tracking stations that are identical but with the sensor oriented in a different direction and the test station where the DUT is placed. In the case that the sensor resolution is the same in X and in Y the stations are simply referred to as tracking stations. **Figure 1** shows the described architecture of the system.

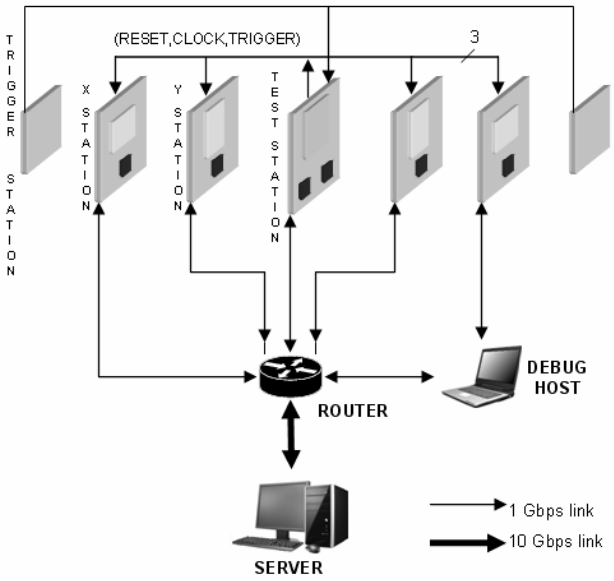

Figure 1 Network based test beam architecture.

The complete system is composed of several *stations*, one router, one server and several peripheral devices that can be plugged into the network and be supported by the server software.

Most data traffic in the system originates from the *stations*, while secondary traffic comes from the server and

from optional peripheral devices. In this manner the data flows from the *station*, or source, to the server, or receiver.

The basic protocol used for the system is UDP/IP while TCP/IP is also supported for secondary applications.

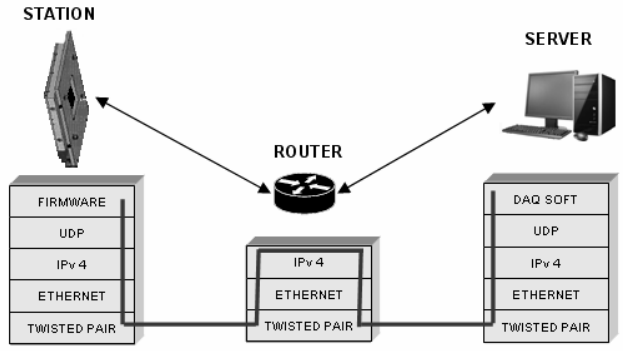

Figure 2 Data flow.

A detailed description on how the data from the *stations* flows through the different layers of the system is shown in **Figure 2**. This is the data path from *station* to server and is responsible for up to 90% of the network traffic.

The application layer is composed by custom made firmware on the *station* end and by custom made software on the server end. All the other network layers use standard protocols.

For the transport layer the UDP protocol was chosen in order to provide faster data transfer, QoS is ensured by the application layer.

Although it's envisioned to use IPv4 as the chosen IP protocol version for the network layer, the system can easily be upgraded for IPv6, although no reason for doing so is foreseen as the router implements a NAT table in order to generate the internal IP address of the network [4].

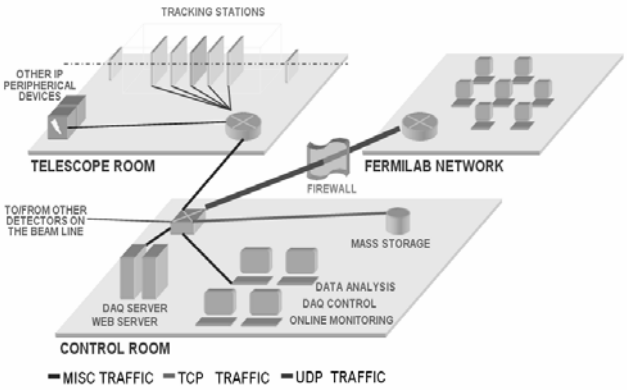

Figure 3 One possible configuration for the network

The data link uses *Ethernet* protocol making it compatible with a wide variety of commercial equipment and making it possible to use on twisted copper pairs in order to run gigabit *Ethernet*.

It is possible to configure the network with many subtle variances without having to redesign software or firmware. In **Figure 3** one possible configuration is shown while in **Figure 1** the bare minimum effective configuration was presented.

Also, it is possible to observe that for a more complete *telescope* system, different computers can run different applications in parallel under the DAQ server and even a second server can be set on the network to be used to provide external control or data access to the web.

III. System Implementation

The system is divided here into three layers as illustrated in **Figure 2**, the hardware, the firmware and the software. Connecting all the layers we have the communication protocol.

The system hardware is comprised of custom made and commercially available hardware. The commercial hardware has no special requirements, its specifications are well within the current available off the shelf hardware, and it consists basically of routers, *Ethernet* cards and computers. The custom made hardware is composed by the tracking *stations*. In order to minimize the number of PCB designs, the *station* design is done in such a way that it can be used as an *X station*, *Y station* or a test *station*. The basic idea of the *station* architecture is illustrated in Figure 4.

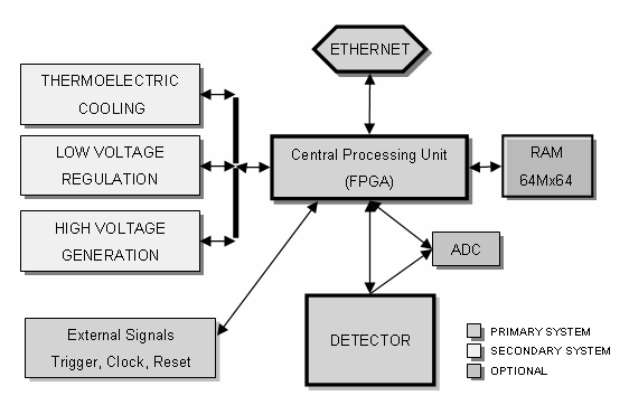

Figure 4 Tracking *stations* architecture.

The *station* is basically composed of an *Ethernet* port that sends and receives data from the CPU. The CPU is implemented on a FPGA and provides the only communication channel that the board has with the network. All operations and data must pass through the CPU. The CPU is shown in **Figure 5** as a top level diagram. The firmware that implements the CPU and the tracker *station* hardware are the two main custom components that exist on the *telescope* side.

The basic task of the CPU is to acquire data from the detector, process the data, store it into the memory and finally send the processed data through the *Ethernet* to the server. Also, it is a task of the CPU to interpret commands received through the network and execute them on the primary or secondary commands processor. The primary commands

processor handles communications intended for the detector, while the secondary commands processor communicates with the various subsystems.

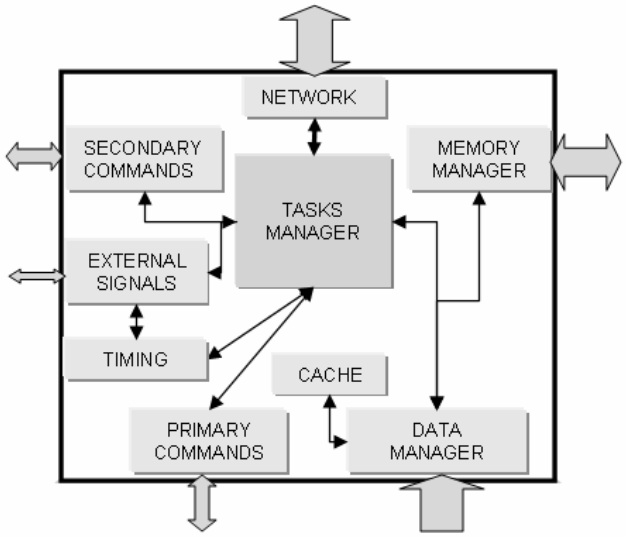

Figure 5 Tracker plane CPU.

When the CPU acquires data from the detector, the data manager sorts the incoming hits by time stamp. If the time stamp is not provided by the detector the CPU stamps the data using the counters located in the timing block. In order to organize the data in this fashion the data manager utilizes a cache block. The sorted data is then packed into 64-bit words and stored in the memory.

The task manager block controls the data flow between blocks. In order to data transfer be efficient, data is presented to the network in form of data blocks.

DATA TYPE	PROTOCOL	ORIGIN
RAW DATA	DETECTOR DATA UP TO 32 BITS	DETECTOR
DATA WORD	64 / 32 / 0 — TIME STAMP | CHIP ID | DETECTOR DATA	DATA MANAGER
UDP FRAME	VARIABLE LENGHT — UDP HEADER | SOURCE PORT | DESTINATION PORT; DATA WORD	NETWORK BLOCK
IP FRAME	VARIABLE LENGHT — IP HEADER | SOURCE IP | DESTINATION IP; UDP FRAME	NETWORK BLOCK
ETHERNET FRAME	VARIABLE LENGHT — ETHERNET HEADER | SOURCE MAC | DESTINATION MAC; IP FRAME	NETWORK BLOCK
COMMUNICATION CHANNEL		

Table 1 Data flow and packaging hierarchy.

The network subsystem has the task of encapsulating the data for the transport, network, and link layer; this concept is better illustrated by **Table 1**.

The data coming from the detector is expected to be, at most, 32 bits long. Otherwise the data manager will fragment it in two data words. Also the board has an optional ADC that can be utilized by detectors with an analog output. The ADC will condition their data into a binary format so the data manager can deal with it.

Another dedicated block is the external signal block that is a communication channel reserved for *inter-station* direct communication, trigger signals, a global reset, and clock distribution.

The custom engineered portion on the server side is the software. Only the DAQ and control components of the software are mentioned here, that is the part that collects and stores the data, provides system and data monitoring tools, and controls the *telescope*. The analysis and database tools are not included in this work, but the software is written in such a way to provide maximum interfacing flexibility.

At the PC application level, the software will employ a design structure based on the concept of shared memory. All communication will flow through the shared memory – facilitated by multiple threads and processes. Exploiting the multi-threading capabilities of C++ will allow these multiple execution paths to access the data in parallel.

For example, within the main server process, there will be threads for receiving data from, and sending commands to, the *telescope* and device under test. Additionally, there will be a thread spawned for each graphical user interface (GUI) that connects to the main server. The only common interface among the threads will be the shared memory.

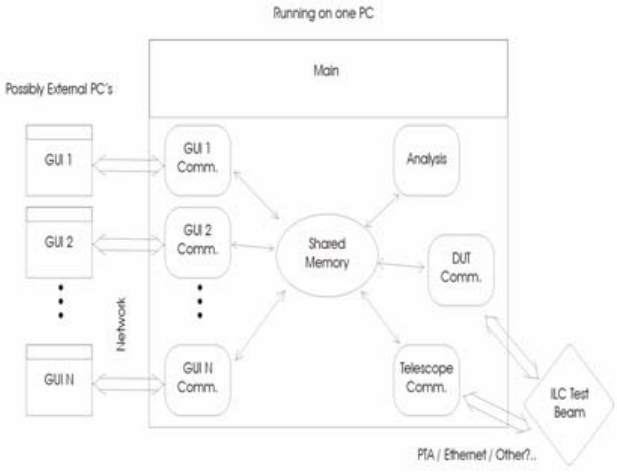

Figure 6 This figure shows the expected structure of the test beam DAQ software. The square-edged blocks are foreseen as processes, while the round-edged blocks are threads.

In this framework there are three end users of the data: the GUI's, the on-line analysis, and the off-line analysis. The GUI in this system is the entity that handles the slow controls and much of the real-time data display and feedback. The on-line analysis is handled by a separate thread, within the main server, and deals with simple calculations and data organization that can be completed on the fly. Finally the off-line analysis, not shown in **Figure 6**, completes the more rigorous computations such as particle track reconstruction and extensive calibration procedures.

The flexibility afforded by such a design will allow the GUI's, the on-line analysis, and the off-line analysis to run seamlessly – whether they are executing on a local machine or

an external machine, as long as they comply with the communication protocol.

The ILC test beam DAQ software will ultimately be judged on how well it can adapt to multiple external users simultaneously accessing data, and dynamic devices under test. For this reason, careful planning must go into every phase of the design.

IV. System Capabilities

The system was designed with high data rate capability, flexibility, reliability, ease of maintenance, and user-friendliness in mind.

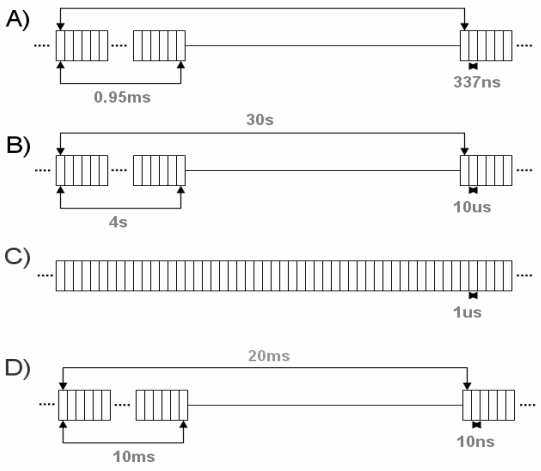

Figure 7 A) ILC foreseen beam structure, B) 2005 CMS testbeam at M-Test/Fermilab, C) Continous beam system capabilitie, D) Pulsed beam system capabilite supposing a occupancy equal to 6 and 5 tracking *stations* and 1Gbps links.

The system high data rate capability is achieved using an onboard memory and a fast link. Initially the system is proposed to be built with 1Gbps links and a 64Mb memory with 64-bits words. Although this is believed to be more than enough for the current test beam requirements, as Figure 7 illustrates, the system can easily be upgraded to 128Mb onboard memory and a 10Gbps link between the router and the server. Considering the initial configuration, which means five *stations* with 64Mb RAM per *station* and 1Gbps links, the system have a continuous readout capability of 160Mbps. A rate of 640Mbps can be achieved for a beam structured as item (D) shows in Figure 7 (at least 10ms of inactivity between spills, spills have at most a 10ns period, and spill train duration is no longer than 10ms).

The system presented here is built with maximum flexibility in mind. Crucial design decisions were made to increase flexibility such as the inclusion of IP based communications, a modular hardware architecture (where many sub-circuits are supported but optional), a firmware based hardware design, and finally a shared memory software approach. Moreover there are many external general purpose pins, connected to the FPGA, on the *telescope* tracking *stations*. The pins provide inputs for external trigger devices and a full control of the system through the network including control of the power supply, temperature and *telescope* positioning.

The system lends itself nicely to debugging. The use of IP protocol allows, for example, to connected a laptop directly into one *station* or between one *station* and the router, enhancing the debugging capabilities of the system.

Figure 3 illustrates how it is straight forward to increase the number of planes that the system operates with. The number of *stations* is basically limited just by the number of router channels and the required system readout speed. This capability comes from the use of a NAT (Network Address Translation) capable router that can provide up to 65,536 different addresses to support extra *stations* or peripherals.

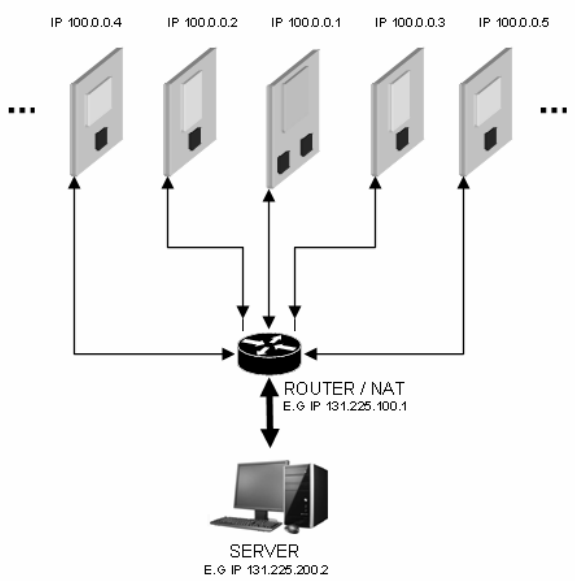

Figure 8 NAT based solution.

Although **Figure 1** illustrates the system configured with 1Gbps/10Gbps links it can comfortable run with 100Mbps or even 10Mbps links depending on the beam structure.

V. Conclusions

This work demonstrated that to meet every ILC test beam foreseen requirements [5], a network based solution is an appropriate approach. It was shown that, with such architecture, a flexible, reliable, and user-friendly system could be realized.

The fact that the proposed architecture makes extensive use of commercial available hardware also decreases its cost and support needed.

The inherent flexibility of using an internet based topology easily justify the small increase on the system complexity, the increase of complexity is transparent for the system designer as well as for the user.

This works provides a road map for the actual implementation of the system, pointing to all the standard

protocols that should be used as well as the basic strategy of the overall system architecture.

ACKNOWLEDGMENTS

The authors thank the members of the Detector Instrumentation Group at Fermi National Accelerator Laboratory who made contributions to the test stand presented in this paper: J. Andresen, S. Bledsoe, G. Cardoso, J. Chramowicz, H. Connor and G. Deuerling. As well as INFN / Milan DAQ members from the CMS/BTeV Test Beam Dario Menasce, Luigi Moroni, Daniele Pedrini.

REFERENCES

[6] L. Uplegger, J. A. Appel, M. Artuso, G. Cardoso, H. P. Cease, G. Chiodini, D.C. Christian, D.A. Cinabro, R. Coluccia, J. Hoff, S. Kwan, S. Magni, A. Mekkaoui, D. Menasce, C. Newsom, V. Papavassiliou, A. Schreiner, M.A. Turqueti, R. Yarema, J.C. Wang , "First look at the beam test results of the FPIX2 readout chip for BTeV silicon pixel detector", IEEE Nuclear Science Symposium and Medical Imaging Conference (NSS / MIC), Rome, Italy, 16-22 Oct 2004. Published in IEEE Trans.Nucl.Sci.53:409-413, 2006.

[7] S. Kwan, J.A. Appel, J.N. Butler, G. Cardoso, H. Cheung, G. Chiodini, D.C. Christian, E.E. Gottschalk, B.K. Hall, J. Hoff, P.A. Kasper, R. Kutschke, A. Mekkaoui, R. Yarema, S. Zimmermann, C. Newsom, A. Colautti, D. Menasce, S. Sala, R. Coluccia, M. Di Corato, M. Artuso, J.C Wang, "Beam test results of the BTeV silicon pixel detector" Nuclear Science Symposium Conference Record, 2000 IEEE Volume 1, Issue , 2000 Page(s):3/49 - 3/53 vol.1.

[8] E. Ramberg , "FNAL Beam Test Facility Status and Plans", on ILC Detector Test Beam Workshop17-19 January 2007.

[9] C. Girerd, S. Gardien, J. Burch, S. Katsanevas, J. Marteau, "Ethernet network-based DAQ and smart sensors for the OPERAlong-baseline neutrino experiment", IEEE Nuclear Science Symposium, 12/111-12/115 vol.2, 2000.

[10] J. F Kurose, K.W. Ross, "Computer Networking: A top-down approach featuring the internet",third ed., Addison-Wesley, p331-p341,2005.